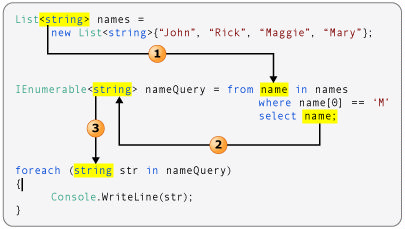
¿Qué es LINQ?

1. Sus significados por sus siglas es Language Integrated Query (Lenguaje Integrado de Consultas).
2. Son un conjunto de herramientas de Microsoft para realizar todo tipo de consultas a distintas de datos como objetos, xmls y base de datos.
3. Surge en el framework 3.5 y pronto coge una gran aceptación en el mundo .NET.
4. En 2008, es incorporado a C# 3.0.
5. Desde su surgimiento estuvo vinculado a herramientas como expresiones lambdas, inferencia de tipos, genericidad y tipos anónimos.

Información general de las expresiones de consultas:

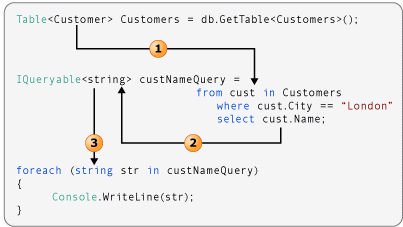
1. Se pueden utilizar para consultar y transformar los datos de cualquier origen de datos habilitado para LINQ.
2. Las expresiones de consulta son fáciles de controlar porque utilizan muchas construcciones de lenguaje C#.
3. Todas las variables de una expresión de consulta están fuertemente tipadas, aunque en muchos casos no es necesario proporcionar el tipo explícitamente porque el compilador puede deducirlo.
4. Una consulta no se ejecuta hasta que no se realiza la iteración a través de la variable de consulta.
5. En tiempo de compilación, las expresiones de consulta se convierten en llamadas al método de operador de consulta estándar según las reglas establecidas en la especificación de C#. Cualquier consulta que se puede expresar con sintaxis de consulta también se puede expresar mediante sintaxis de método. Sin embargo, en la mayoría de los casos, la sintaxis de consulta es más legible y concisa.
6. Como regla al escribir consultas LINQ, se recomienda utilizar la sintaxis de consulta siempre que sea posible y la sintaxis de método cuando sea necesario. No hay diferencias semánticas ni de rendimiento entre estas dos formas diversas. Las expresiones de consulta suelen ser más legibles que las expresiones equivalentes escritas con la sintaxis de método.

Consultas que no transforman los datos de origen:

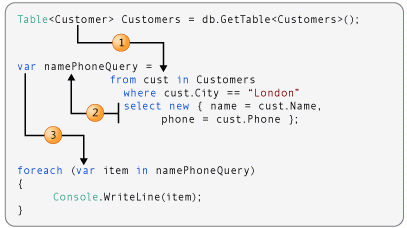


1. El argumento de tipo del origen de datos determina el tipo de la variable de rango.
2. El tipo del objeto que está seleccionado determina el tipo de la variable de consulta. Aquí, *name* es un *string*, por tanto, la variable de consulta es *IEnumerable<string>.*
3. La variable de consulta se procesa en iteración en la instrucción *foreach*. Dado que la variable de consulta es una secuencia de *string*, la variable de iteración también es un *string*.

Consultas que transforman los datos del origen:



1. El argumento de tipo del origen de datos determina el tipo de la variable de rango.
2. La instrucción *select* devuelve la propiedad *Name* en lugar del objeto *Customer* completo. Dado que *name* es un *string*, el argumento de tipo de *custNameQuery* es string y no *Customer*.
3. Dado que *custNameQuery* es una secuencia de *string*, la variable de iteración del ciclo *foreach* también debe ser *string*.

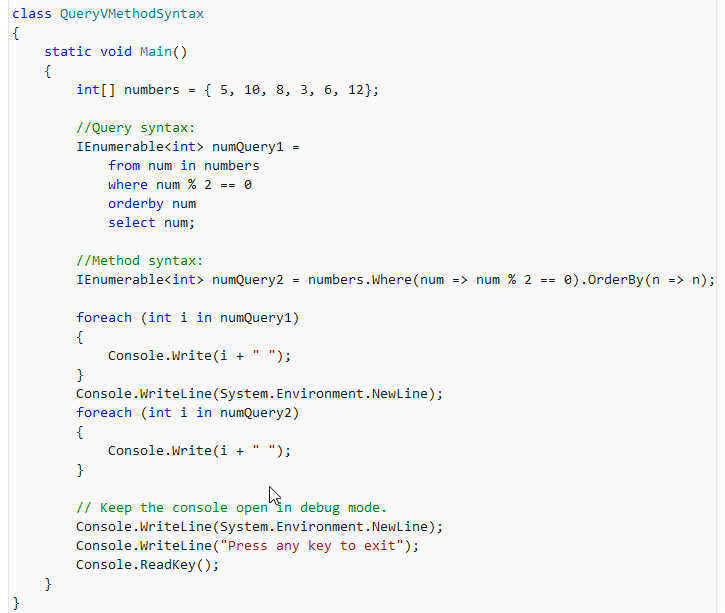


1. El argumento de tipo del origen de datos siempre es el tipo de la variable de rango de la consulta.
2. Dado que la instrucción *select* genera un tipo anónimo, la variable de consulta debe declararse implícitamente mediante *var*.
3. Dado que el tipo de la variable de consulta es implícito, la variable de iteración del bucle *foreach* también debe ser implícita.

Sintaxix de consultas y sintaxis de métodos en LINQ:

La mayoría de las consultas de la documentación introductoria de Language Integrated Query (LINQ) se escribe con la sintaxis de consulta declarativa de LINQ. Pero la sintaxis de consulta debe traducirse en llamadas de método para .NET Common Language Runtime (CLR) al compilar el código. Estas llamadas de método invocan los operadores de consulta estándar, que tienen nombres tales como *Where*, *Select*, *GroupBy*, *Join*, *Max* y *Average*. Puede llamarlas directamente con la sintaxis de método en lugar de la sintaxis de consulta.

La sintaxis de consulta y la sintaxis de método son idénticas desde el punto de vista semántico, pero muchos usuarios creen que la sintaxis de consulta es mucho más sencilla y fácil de leer. Algunos métodos deben expresarse como llamadas de método. Por ejemplo, debe usar una llamada de método para expresar una consulta que recupera el número de elementos que cumplen una condición especificada. También debe usar una llamada de método para una consulta que recupera el elemento que tiene el valor máximo de una secuencia de origen.



Por qué las consultas LINQ son tan eficientes:

* Tienen la capacidad de crear nuevos tipos, esto se realiza en la cláusula *select*.
* Pueden combinar varias secuencias de entrada en una sola secuencia de salida que tiene un tipo nuevo.
* Pueden crear secuencias de salida cuyos elementos estén formados por una o varias propiedades de cada elemento de la secuencia origen.
* Pueden crear secuencias de salida cuyos elementos estén formados por los resultados de las operaciones realizadas en el origen de datos.
* Pueden crear secuencias de salida en un formato diferente.

Tipos anónimos:

Un tipo anónimo se construye por el compilador y el nombre del tipo solo está disponible para el compilador. Los tipos anónimos son una manera cómoda de agrupar un conjunto de propiedades temporalmente en un resultado de consulta sin tener que definir un tipo con nombre independiente. Los tipos anónimos se inicializan con una nueva expresión y un inicializador de objeto, como se muestra aquí:

